02-1526

LA-UR-00-5693-REV1

Approved for public release; distribution is unlimited.

c.1

Title: GIPPER USER MANUAL

Author(s): Billy J. Archer, Robert E. H. Clark, Christopher J. Fontes, and Honglin Zhang

Submitted to:

This page is intentionally blank.

GIPPER USER MANUAL

Billy J. Archer, Robert E. H. Clark, Christopher J. Fontes, and Honglin Zhang

Los Alamos National Laboratory
P.O. Box 1663, MS F663
Los Alamos, New Mexico 87545

Abstract

GIPPER is part of the theoretical atomic physics and applications code family at Los Alamos. The GIPPER code uses the atomic structure data calculated by the CATS or RATS codes to calculate ionization cross sections. Photoionization, electron collisional ionization, and autoionization cross sections can be calculated using configuration average, multiplet terms, or fine structure levels. At publication time all relativistic calculations are possible in the configuration average approximation. Relativistic fine structure calculations are possible only for photoionization. This paper describes the use of GIPPER.

I. Introduction

This manual describes how to use the General Ionization Processes in the Presence of Electrons and Radiation (GIPPER) code. GIPPER is part of the theoretical atomic physics and applications code family developed at Los Alamos National Laboratory [1, 2].

The GIPPER code is used to calculate cross sections for ionization processes produced by electrons and photons. The GIPPER code uses the atomic structure data calculated by the CATS (non-relativistic) [3] or RATS (relativistic) codes [1]. The GIPPER code can calculate cross sections for electron collisional ionization, photoionization, and autoionization in both the non-relativistic and relativistic approximations [4, 5]. For electron collisional ionization the non-relativistic cross sections can be calculated using scaled hydrogenic theory, binary encounter theory, or the distorted wave approximation. In relativistic electron collisional ionization the cross sections can be calculated using scaled hydrogenic theory or the distorted wave approximation. Non-relativistic photoionization cross sections can be calculated with distorted waves or Gaunt factors. Relativistic photoionization results can be calculated with distorted waves or via fit parameters. Autoionization processes are always calculated with distorted waves. Fine structure mixing of target states is possible in all non-relativistic calculations and in relativistic photoionization calculations.

The X-5 Physical Data Team maintains GIPPER. Contacts are Honglin Zhang or Christopher Fontes.

This manual first lists the input commands available in GIPPER by function. Several examples of input control files are shown next. The largest section of the manual is a brief description of each command in alphabetical order, the allowed parameters for each command, and the default parameter. Next is a description of how to build the code. The last section shows how to run the code and gives a brief description of how to view the output.

II. Commands By Function

Type *help command* for information on any command.

Code action commands

end	go	help

I/O commands

blocksiz	files	filesize	infile	io_size
num_recs	outfile	outfiled	outfilep	outfiler

Material definition commands

chpar	ion	seq

Display commands

cfg	levels	levelsl	terms	termsl
tty	wf			

Transition selection commands

setc	setl	setle	sett

Mode commands

ai	bien	bound	cfge	core
csmooth	dbug	dw	emode	eunits
expot	fitpi	free	ftype	integral
mix	mixmode	mode	partl	pig
points	potl	print	saime	setcip
setz	smooth	spifail	spime	tally

wraim	wrame	wrcfg	writebe	writefe
writeprd	xqion			

Grid commands

e	elin	elog	hint	mshno
nixc	nixx	npphc	x	xgauss
xlin	xlog			

Numerical parameter commands

abserr	eps001	eps20	epsde	epseig
epsporb	ildel	lbmx	nie	ntbien
refcfg	reflev	refterm	relerr	tlim

III. Example Input Control Files

Four example control files are shown below. The control files are just ASCII text files whose use is specified by the infile command.

IGIP2

IGIP2 is a non-relativistic configuration average calculation of collisional ionization using the distorted wave approximation, with photoionization turned off. It reads CATS files with the default names group1 and group2. Output is stored or appended onto a IPCRESS format file named gipout2.

```
mode cfg
outfile gipout2
xlog 21 1.01 100.
pig f
dw t
setc 1 *
go
end
```

IRGIPN1

IRGIPN1 reads in data files rgrpn1 and rgrpn2, writes or appends the result to a PARADISE format file gipoutr. It is carrying out a configuration average, photoionization calculation.

```
mode cfg
files rgrpn1 rgrpn2
outfilep gipoutr
xlog 21 1.01 100.
go
end
```

IGIP.AI

IGIP.AI reads files fe16f and fe17f, writes the result to an IPCRESS file gfe16f, first deleting any previous version. Photoionization is turned off and autoionization is turned on for this configuration average calculation.

```
files fe16f fe17f
outfiled gfe16f
mode cfg
pig f
ai t
go
end
```

IGIPF

IGIPF reads in data files rgrpn1 and rgrpn2, writes the result to an PARADISE format file gipout, first deleting any previous version. It is carrying out a fine structure, photoionization calculation.

```
files rgrpn1 rgrpn2
outfiler gipout
xlog 21 1.01 100.
go
end
```

IV. Command Details

A brief summary of each command and its arguments are below. Some commands have alternate command names that do the same thing. These alternate commands are listed with the primary command for backward compatibility, but are not given separate paragraphs.

If a command is issued to GIPPER with no argument the current value is returned and the program waits for user input of a new command.

abserr

The abserr command sets the requested absolute error for numerical integration routines. The command structure is

ABSERR real

where real is the desired limit. The default is 0.0001.

ai

The ai command turns the autoionization calculation on or off. The command structure is

AI logical

where logical can be *t*, *true*, *f*, or *false*. The default value is *false*.

Alternate command forms are autoion, auto, or aa.

bien

The bien command turns the binary encounter collisional ionization calculation on or off. The command structure is

BIEN logical

where logical can be *t*, *true*, *f*, or *false*. The default value is *false*.

blocksiz

The blocksiz command sets the maximum size in words of blocks of data on the output file. The blocksiz command must be issued before the outfile command. The command structure is

BLOCKSIZ integer

where integer is the desired size. The default value is 50,000 words.

bound

The bound command is used to choose the type of radial wave function for bound states. The command format is

BOUND string

where string is *hf* for CATS/RATS wave functions or *hyd* for internally generated hydrogenic wave functions. The *hyd* option should only be used with non-relativistic calculations. The default is *hf*.

cfg

The cfg command is used to list the configurations in the current problem. The command format is

CFG

with no options.

cfge

The cfge command sets the method by which configuration average energies are determined. The command structure is

CFGE string

where if string is *configs* or *cfg* the configuration average energies are read from the structure files. If string is *levav* or *levels* the fine structure energy levels, which usually include the effect of configuration interaction, are averaged for each configuration. The default value is *configs*.

An alternate form of the command is *ecfg*.

chpar

A list of materials (Z, ionicity, sequence number) currently on a file can be viewed at the terminal using the command

CHPAR fname

where fname is the name of the file to be checked. If fname is omitted, the current output file name is used. The default file is *gout*.

An alternate command form is *checkout*.

core

The core command determines whether or not the unxecited core electrons will contribute to the cross section. The command structure is

CORE logical

where logical can be *t*, *true*, *f*, or *false*. The default is *false*.

csmooth

Turns on/off smoothing of collision ionization cross sections. Command format is

CSMOOTH logical

where logical can be *t*, *true*, *f*, or *false*. Default is false.

dbug

The dbug command allows printing of various parameters for debugging. The command structure is

DBUG integer option

where integer is the flag number between 1 and 100, and option can be *t*, *true*, *f*, or *false*. Only the first few flags actually are used. The default is false for all flags.

dw

The dw command turns the distorted wave collisional ionization calculation on or off. The command structure is

DW logical

where logical can be *t*, *true*, *f*, or *false*. The default is false. Also see the free command for additional capabilities.

e

The e command is used to type in initial/final energies for electrons or photons. The command format is

E e1 e2 e3 . . .

where e1, e2, etc are desired energies in the units set by the eunits command. Up to 100 energies may be used. Making the last character on the line an ampersand and continuing the energies on the next line without the command name continues the command line. Also see the emode command.

elin

The elin command is used to set up a linear grid of initial/final energies for electrons or photons. The command format is

ELIN ne el eu

where ne is the number of energies, el is the lower limit and eu is the upper limit. The energy units are set by the eunits command. Up to 100 energies may be used. Also see the emode command.

elog

The elog command is used to set up a logarithmic grid of initial/final energies for electrons or photons. The command format is

ELOG ne el eu

where ne is the number of energies, el is the lower limit and eu is the upper limit. The energy units are set by the eunits command. Up to 100 energies may be used. Also see the emode command.

emode

The emode command sets the energy mode of the input energies as either initial (impact) or final (ejected). The command format is

EMODE string

String can have values of either *initial* or *final.* Default is *initial.* For example, if emode is *final* and photoionization is being calculated, the energies are those of the ejected electron. For the case of collisional ionization, a value of emode of *final* allows the user to input values for the sum of the energies of the two outgoing electrons. This command affects both absolute and threshold energy commands.

end

The end command terminates the GIPPER run and closes all files. The command format is

END

with no arguments.

eps001

Set eps001 to control convergence of the first outgoing electron partial wave functions in a distorted wave ionization calculation. Command format is

EPS001 real

where any real is allowed. Default is 0.0001.

eps20

Set eps20 to control convergence of the incoming electron partial wave functions in a distorted wave ionization calculation. Command format is

EPS20 real

where any real is allowed. Default is 0.01.

epsde

The epsde command allows control over the recalculation of matrix elements between levels with different transition energies. The command structure is

EPSDE eps

where eps is the desired limit. If all levels are within eps of the configuration average, the calculation will use one energy. The default is 0.001, interpreted as 0.1%.

epseig

The epseig command sets a lower limit on the magnitude of the eigenvector components (mixing coefficients) of the initial and final level that will be retained. The command structure is

EPSEIG eps1 eps2

where eps1 is the limit for the initial level and eps2 is the limit for the final level. If eps2 is omitted, it is set to the value of eps1. Any basis state that contributes less than the appropriate limit to the level will be ignored. The default values are 0.001 for both eps1 and eps2. Also see the mix command.

epsporb

The epsporb command sets the value for determining that a bound state wave function has gone to zero. The command structure is

EPSPORB eps

where eps is the limit. The default value is 1.e-99.

eunits

The eunits command is used to specify the units of input energies. The command format is

EUNITS option

where option may be *x* for threshold units, *ev* for electron volts, *ryd* for rydbergs or *au* for atomic units (Hartrees). The default is threshold units. Threshold units are energies relative to the relevant transition energies, all others are absolute energies.

expot

The expot command is used to control the method of calculating the exchange potential. The command format is

EXPOT string

where string may be *f* to turn off the exchange potential entirely, *sce*, *hee*, *scfee*, *sofege*, *aafege*, *slater23*, or *hx*. These abbreviations are those of Riley and Truhlar (J. Chem. Phys. v63,2182 (1975)). The default is *sce*.

files

The files command tells GIPPER which file names to read for the structure data. The command format is

FILES fname1 fname2

where fname1 and fname2 are the file names for the two ionization stages. Two file names must be entered unless the initial target state is hydrogenic in which case only one filename is input. The default file names are group1 and group2.

filesize

The filesize command determines the maximum file size in words of individual output files. This command effectively sets the size of data record portion of file, determined by the filesize minus the sizes of the prefix and index. The filesize command must be issued after the num_recs command and before the outfile command. The command structure is

FILESIZE integer

where integer is the desired size in words. The default is 5000000.

fitpi

The command fitpi controls whether fitting is used for relativistic photoionization cross sections. If fitting is used the code reads in photoionization cross sections for each subshell, then fits to a specific transition energy for each configuration to configuration transition. The photoionization cross sections are read from the file named iphxs that is made by the stand alone code rdwpi. The command format is

FITPI integer

where integer can be *0* or *1*. If 0 do not use fitting, if 1 use fitting. Default is 0.

free

The free command is used to select the type of radial wave functions for the free electron. The command format is

FREE string

where string is *dw* for a distorted wave calculation, *coul* for Coulomb functions, or *gaunt* for Gaunt factor cross sections. The last two options are available only for non-relativistic calculations. The default is distorted wave.

ftype

Choose the type of distorted wave free wave function for collisional ionization. Command format is

FTYPE string

where string can be *mann* or either *unmod* or *nonmod*. Default is *unmod*.

go

The go command causes GIPPER to begin the ionization calculation. The command format is

GO

with no arguments.

help

The help command is used to give information on GIPPER commands. The command format is

HELP string

where string is a command for which information is requested. The help command with no arguments will list all available commands.

hint

The hint command is used to determine the initial step size in the radial mesh. The command structure is

HINT real

where real is the desired initial step size in atomic units. A default value is calculated as $\frac{HMIN}{2}\left(\frac{3\pi}{4}\right)^{2/3}\frac{1}{Z^{1/3}}$ if no value is given for hint. For a non-relativistic calculation $HMIN = 0.0025$ and for a relativistic calculation $HMIN = 3.125\times10^{-4}$. Giving a value of less than 1×10^{-10} will reset hint to the default calculated value.

ildel

Controls the spread between the initial and final electron orbital-angular momentum coupling in collisional ionization. Command format is

ILDEL integer

where integer must be between 0 and 30. Default is 30.

infile

The infile command is used to transfer control from the terminal to a set of instructions in a text file. The command format is

INFILE fname

where fname contains the commands. The default is to get input from the terminal.

integral

The integral command selects the integration scheme to be used for numerical integration. The command format is

INTEGRAL string

where string may be *leeint* for Lee Collins' subroutine, *bobint* for R. E. H. Clark's subroutine or *weddle* for a UCL subroutine. The default is *weddle*.

ion

The ion command is used to select a particular material from the input files. The command format is

ION z ion seq1 seq2

where z is the atomic number, ion is the ionicity in spectroscopic notation for the initial target state and seq1 and seq2 are the sequence numbers for the initial and final ion. If seq1 and seq2 are omitted the code will use the first entry found. If the word *all* is the only argument the code will calculate data for all possible ion stages from the two input files. The default is to use the first sequence found on each file.

Alternate forms of the command are *zions* or *zion*.

io_size

The io_size command is used to set the number of processes in each MPI I/O communicator for parallel calculations. The command format is

IO_SIZE integer

where integer is the number of processes per I/O communicator. The number of processes per communicator must be less than or equal to the total number of processes. The best size for the I/O communicator is machine architecture dependent. A default value is set depending on the total number of processors, assuming a SGI architecture.

lbmx

The lbmx parameter sets the upper limit on the ℓ-value of the impact free electron in a distorted wave calculation. The command structure is

LBMX integer

where integer is the desired value. The default value is 20.

levels

The levels command is used to list the levels in the current problem. The command format is

LEVELS

with no options.

levelsl

The levelsl command is similar to the levels command except that it lists block and script quantum numbers for each shell. The command structure is

LEVELSL

with no options.

mix

The mix command determines whether or not target state mixing will be included. The command structure is

MIX logical

where logical is *t*, *true*, *f*, or *false*. The default value is *true*. Also see the epseig command.

mixmode

Select the factorization method for calculating cross sections. Command format is

MIXMODE string

where string can be *old* or *new*. Default is new.

mode

The mode command is used to select the level at which cross section data is to be calculated. The command format is

MODE string

where string may be *level* or *levels* for fine structure levels, *term* or *terms* for LS terms or *cfg*, *cfgs*, *config*, or *configs* for configuration average. Target state mixing is only available in the level mode. The default is levels.

mshno

The mshno command sets the number of points per block in the radial mesh. The command structure is

MSHNO integer

where integer is the desired number between 1 and 55. The default value is 40.

nie

The nie command sets the number of transition energy points at which calculations will actually be carried out. Interpolation between these points is then used for individual transitions. The command structure is

NIE integer

where integer is the desired number of points. The default is 5.

nixc

Set the number of mesh points for relativistic continuum functions. Command format is

NIXC integer

where integer must be greater than zero. Default is 15,000. This variable will be increased automatically by the code when necessary. Therefore, this command is not particularly useful for large-scale calculations, but might be useful for debugging.

nixx

The nixx command is used to set the number of blocks in the radial mesh. The command structure is

NIXX integer

where integer is the desired number of blocks, between 1 and 55. The default is 55.

npphc

The npphc command controls the number of points per half cycle of the continuum function in the asymptotic region. The command structure is

NPPHC integer

where integer is between 1 and 55. The default is 8.

ntbien

The ntbien command sets the number of points in t-space to be calculated in a binary encounter calculation. The command structure is

NTBIEN integer

where integer is between 1 and 2000. The default is 150.

An alternate form of the command is *nt*.

num_recs

The num_recs command sets the maximum number of records on each member of a output family file. Effectively this sets the size of the index. The num_recs command must be issued before the outfile or filesize commands. The command structure is

NUM_RECS integer

where integer is the number of records. The default is 7680.

outfile

The default output file is a IPCRESS file named 'gout'. The outfile command renames the output IPCRESS format file. The outfile command must be issued after the filesize and num_recs commands. The command format is

OUTFILE fname

where fname is the desired name. Output can be suppressed by a filename of *f* or *false*. If a family of files will be used, fname must be six characters or less. If running in parallel the filename must be four characters or less. If a file of name fname already exists, the new output is appended to the file. However, if the file fname is a family file, execution is stopped because appending to a family file is not allowed.

If the output filename is specified as base then the format of the family filenames are:

- Serial: base, basea, baseb, ...
- Parallel: base01, base01a, base01b, ..., base02, base02a, base02b, ...

where a, b, c and so on designate family file members and 01, 02, 03 and so on designate I/O sub-communicators.

Alternate forms are *outfilei* or *outfilen*.

outfiled

The same as outfile except this command deletes any preexisting file named fname. During serial runs it also deletes any related family files such as fnamea, fnameb,etc. During parallel runs it deletes old I/O communicator files and related family files, i.e. fname01, fname01a, fname02, etc. The command format is

OUTFILED fname

where fname is the desired name. See the outfile command for the file name format.

outfilep

The outfilep command causes the output to be in PARADISE format. The command format is

OUTFILEP fname

where fname is the desired name. The default filename is named 'gout'. Output can be suppressed by a filename of *f* or *false*. If a family of files will be used, fname must be six characters or less. If running in parallel the filename must be four characters or less. If a file of name fname already exists, the new output is appended to the file. However, if the file fname is a family file, execution is stopped because appending to a family file is not allowed. See the outfile command for the file name format.

outfiler

The same as outfilep except this command deletes any preexisting file named fname. During serial runs it also deletes any related family files such as fnamea, fnameb,etc. During parallel runs it deletes old I/O communicator files and related family files, i.e. fname01, fname01a, fname02, etc. The command format is

OUTFILER fname

where fname is the desired name. See the outfile command for the file name format.

partl

Add-on or leave off extrapolation for distorted wave collisional ionization calculation. Command format is

PARTL logical

where logical can be *t*, *true*, *f*, or *false*. Default is *true*.

pig

The pig command turns photoionization generation on or off. The command structure is

PIG logical

where logical can be *t*, *true*, *f*, or *false*. Default is *true*.

points

Select whether to calculate distorted wave ionization cross sections at each explicitly input impact energies, or to calculate via interpolation of cross sections calculated at six predetermined energies that are hardwired in subroutine DWARSPP. If POINTS is true, an explicit calculation is always carried out. If POINTS is false then the interpolation scheme is used *provided* that one of the following is true: (1) the number of input energies is greater than six OR (2) the first and/or last input energy is outside of the hardwired range specified in routine DWARSPP (x=1.00999 - 10.001). If neither of these

provisions is satisfied, an explicit calculation is carried out (even though the value of POINTS remains *false*). Command format is

POINTS logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*. Also see the e or x command families for entering energies.

potl

The potl command selects the potential, either Dirac-Fock-Slater (dfs) or Mann, for relativistic continuum wavefunctions. Command format is

POTL integer

where integer can be 0 for *dfs* or 1 for *Mann*. Default is 0.

print

The print command turns on or off the creation of diagnostic files picross, xqout, bienout, dwout, and aiout. The command structure is

PRINT logical

where logical is *t*, *true*, *f*, or *false*. The default is *false*.

An alternate form of the command is *file*.

refcfg

The refcfg command sets a reference configuration for listing energies with the cfg command. The command structure is

REFCFG integer

where integer is the index of the desired configuration. This only affects the listing of configurations; cross section calculations are not affected. The default is 1.

An alternate form of the command is *cfgref*.

reflev

The reflev command sets a reference level for listing energies with the levels command. The command structure is

REFLEV integer

where integer is the index of the desired level. This only affects the listing of levels; cross section calculations are not affected. The default is 1.

An alternate form of the command is *reflevel*.

refterm

The refterm command sets a reference term for listing energies with the terms command. The command structure is

REFTERM integer

where integer is the index of the desired term. This only affects the listing of terms; cross section calculations are not affected. The default is 1.

An alternate form of the command is *termref*.

relerr

The relerr command sets the requested relative error for numerical integration routines. The command structure is

RELERR real

where real is the desired limit. The default is 0.001.

saime

The saime command specifies whether all autoionization matrix elements are to be calculated at one time and saved or not. The command structure is

SAIME option

where option can be *t*, *true*, *f*, or *false*. The default is *true*. Saving the matrix elements increases storage requirements but decreases running time.

seq

The seq command is used to set a sequence number for output data. The command format is

SEQ integer

where integer is the desired number. The default is 1.

setc

The setc command is used to select transitions common to a given configuration to configuration transition. The command format is

SETC i1 i2

where i1 and i2 are a desired lower and upper configuration index as from the CFG command. Either i1 or i2 may be an asterisk to denote all configurations. For example SETC 1 * would select transitions from all levels of configuration 1 to all other levels including the levels of configuration 1. Either i1 or i2 may be a number sign (#) to

denote all configurations but excluding transitions within the same configuration. Default is i1 = *, i2 = *.

An alternate form of the command is *setcfg*.

setcip

Set parameters for collisional ionization. Command format is

SETCIP

There are no arguments, but it calls a routine to interactively set the following parameters that have the indicated defaults:

iejdz = 2, iejdr = 2, iejdrx = 2, iejxz = 1, iejxr = 1, iejxrx = 2, iscdz = 1, iscdr = 1, iscdrx = 1, iscxz = 2, iscxr = 2, iscxrx = 2.

They can be set to either 1 or 2.

setl

The setl command is used to select specific level to level transitions. The command format is

SETL i1 i2

where i1 and i2 are level indices as from the LEVELS command that give the lower and upper levels of the transition. Either i1 or i2 may be an asterisk to denote all levels. For example SETL 1 * would select transitions from level 1 to all other levels. Either i1 or i2 may be a number sign (#) to denote all levels but excluding transitions within the same level. Defaults are i1 = * and i2 = *.

An alternate form of the command is *setlev*.

setle

The setle command is used to type in energies (in eV) for selected levels. The command format is

SETLE i1 i2 i3 . . .

where i1, i2, etc is a list of level indices whose energies are to be replaced. To replace all energies use SETLE *. After the command is issued, GIPPER will list each selected level with its original energy. You may type in a new energy or use a carriage return to leave the energy as is.

sett

The sett command is used to select transitions common to a given term to term transition. The command format is

SETT i1 i2

where i1 and i2 are term indices as from the TERMS command that give the lower and upper term index of the transition. Either i1 or i2 may be an asterisk to denote all terms. For example SETT 1 * would select transitions from all levels of term 1 to all other levels including the levels of term 1. Either i1 or i2 may be a number sign (#) to denote all terms but excluding transitions within the same term. Default is i1 = * and i2 = *.

An alternate form of the command is *setterm.*

setz

The setz command allows choosing the method of computing the residual charge of the initial ion for distorted wave, collisional ionization calculations. The command structure is

SETZ string

where string is *normal* or either *samp* or *sampson* to make the residual charge of both ion stages the same. The default is *normal.*

smooth

Flag whether to smooth out high-energy photoionization cross sections. Command format is

SMOOTH logical

where logical can be *t, true, f,* or *false.* Default is *true.*

spifail

Skip photoionization from autoionizing levels. The command format is

SPIFAIL logical

where logical can be *t, true, f,* or *false.* Default is *false.*

spime

The spime command specifies whether all photoionization matrix elements are to be calculated at one time and saved or not. The command structure is

SPIME option

where option can be *t, true, f,* or *false.* The default is *true.* Saving the matrix elements increases storage requirements but decreases running time.

tally

The tally command specifies whether or not to call sampling routines for creation of a file for the Tally code. The command structure is

TALLY logical

where logical may be *t*, *true*, *f*, or *false*. The default is *false*.

terms

The terms command is used to list the terms in the current problem. The command format is

TERMS

with no options.

termsl

The termsl command is similar to the terms command except that it lists block and script quantum numbers for each shell. The command structure is

TERMSL

with no options.

tlim

The tlim command sets an upper limit in the binary encounter integration over the variable t. The command structure us

TLIM integer

where integer is the desired upper limit. The default is 500. There are no limits on tlim.

tty

The output data can be written to the terminal using the tty command. The command format is

TTY logical

where logical can be *t*, *true*, *f*, or *false*. The default is *false*.

wf

Flag for reading in all wavefunctions at once or one pair at a time. Command format is

WF string

where string can be *all* or *one*. Default is *one*. Reading in all wavefunctions at once increases storage requirements but decreases running time.

wraim

Control writing of the radial dipole matrix elements for autoionization to the file aime. Command format is

WRAIM logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*.

wrame

The wrame command controls the writing of radial matrix elements to the file rameout. The command structure is

WRAME logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*.

wrcfg

Select whether to write configuration data to the output file. Command format is

WRCFG logical

where logical can be *t*, *true*, *f*, or *false*. Default is *true*.

writebe

The writebe command controls the writing of bound electron wave functions to the file tape8. The command structure is

WRITEBE logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*.

writefe

The writefe command controls the writing of free electron wave functions to the file tape8. The command structure is

WRITEFE logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*.

writeprd

The writeprd command controls the writing of product functions to the file tape8. The command structure is

WRITEPRD logical

where logical can be *t*, *true*, *f*, or *false*. Default is *false*.

x

The x command is used to type in initial/final energies for electrons or photons in threshold units. The command format is

X x1 x2 x3 . . .

where x1, x2, etc are desired energies. Up to 100 energies may be used. Making the last character on the line an ampersand and continuing the energies on the next line without the command name continues the command line. Also see emode command.

xgauss

Control the number of gauss points in dwion. Command format is

XGAUSS real

where real has no limits. Default is 2.01.

xlin

The xlin command is used to set up a linear grid of initial/final energies for electrons or photons in threshold units. The command format is

XLIN nx xl xu

where nx is the number of energies, xl is the lower limit and xu is the upper limit. Up to 100 energies may be used. Also see emode command.

xlog

The xlog command is used to set up a logarithmic grid of initial/final energies for electrons or photons in threshold units. The command format is

XLOG nx xl xu

where nx is the number of energies, xl is the lower limit and xu is the upper limit. Up to 100 energies may be used. Also see emode command.

xqion

The xqion command turns the fit to scaled hydrogenic collisional ionization on or off. The command structure is

XQION logical

where logical is *t*, *true*, *f*, or *false*. The default is *false*.

Alternate forms are *shit* or *shitfit.*

V. Building GIPPER

GIPPER is built via a makefile using the GNU utility gmake.

To build the code using default settings type: g*make gipper*. For non-default builds see the section below on Build Options.

To clean up the directory, removing most of the files made during compilation, type: *gmake clean*. To clean up all the files type: *gmake distclean*.

To create a tar file with the source code and related files type: *gmake tar*.

When moving between machines the following parameters in the makefile may need to change: MPI_LIB, LIBS, MPI_INCLUDE, and INCLUDES. These tell the compiler and loader where to find the libraries required when building the code.

Build Options

There are four optional parameters that can be given to gmake: DEBUG, 64, MPI, and RENO. An example of their use is:

gmake DEBUG=false 64=true MPI=true RENO=false gipper

which turns debugging off, turns 64 bit addressing on, turns MPI on, and turns RENO off.

DEBUG

The DEBUG parameter controls the debugging and optimization levels used to build the code. The options are:

- "DEBUG=true" – builds code with –g, debugging on.
- "DEBUG=none" – builds code with –g0 and –O0, no optimization or debugging.
- "DEBUG=false" – builds production version of code, -O2 –OPT:Olimit=0.

Default is "DEBUG=true".

64

The 64 parameter controls 64-bit addressing. The options are:

- "64=true" – build code with –64, 64-bit addresses.
- "64=false" – build code with –32, 32-bit addresses.

Default is "64=false".

MPI

The MPI parameter controls inclusion/exclusion of MPI commands by the preprocessor. The options are:

- "MPI=true" – build parallel code by including MPI commands.
- "MPI=false" – build serial code, exclude MPI commands.

Default is "MPI=false".

RENO

The Reno parameter determines if the code is linked through libmyf90.a or the Reno version of the library. The options are:

- "RENO=true" – use the Reno library.
- "RENO=false" – do not use the Reno library.

Default is "RENO=false". Turning Reno on forces MPI false and DEBUG false.

VI. Running GIPPER

GIPPER is controlled by commands received via standard input. This means GIPPER can be run interactively from the command-line, i.e. the user types in a series of commands. However, typically the commands are put into an ASCII text file that is piped to standard input.

There are three major modes for running GIPPER: from the command-line, submitted as a batch job, or in a debugger.

GIPPER can be built as either a serial code that runs on only one processor or as an MPI code which runs on multiple processors.

At LANL there are two major classes of computers, large production machines with load sharing facility (LSF) and those without.

Table 1 gives the commands needed to run GIPPER in the various permutations of the code and machines. Several assumptions are made:

- A control file, named cntrl, is being used.
- The user asks for a maximum of four processors (llogin –n 4) for interactive jobs.
- Two processors are used for MPI runs.
- The debugger is totalview.

Table 1. Commands to run GIPPER.

	Serial	MPI
Command-line - LSF	Llogin Gipper < cntrl	Llogin –n 4 Mpirun –np 2 ./gipper Infile cntrl
Command-line – non-LSF	Gipper < cntrl	Mpirun –np 2 ./gipper
Batch – LSF	Llogin Bsub –W 4 'gipper < cntrl'	Llogin Bsub –n 2 –W 8 mpirun './gipper < cntrl'
Debug – LSF	Llogin Totalview gipper & In totalview set the input file for stdin, using either < or control-A depending on the totalview version.	Llogin –n 2 MPT 1.4.0.3 and lower: Totalview mpirun –a –np 2 ./gipper & MPT 1.5 and higher: Totalview pam –a –mpi –auto_place 2 ./gipper & In totalview set the input file for stdin, using either < or control-A depending on the totalview version.
Debug – non-LSF	Totalview gipper & In totalview set the input file for stdin, using either < or control-A depending on the totalview version.	Mpirun –np 2 –tv ./gipper & In totalview set the input file for stdin, using either < or control-A depending on the totalview version.

The principal output from GIPPER is a family of binary files that are in either PARADISE or IPCRESS format. The general format of the files is a prefix, an index, and the data. Specific information on the format of PARADISE files can be found in reference [6] and of IPCRESS files in reference [7].

Both format files can be read using utility codes BREAD, BCOMP, or READBYTE. The BREAD code displays the data on the file. The BCOMP code compares the data on two similarly formatted files. Neither BREAD nor BCOMP directly examine the prefix or index data. The READBYTE code can show all the information on a file, including the prefix and index.

Post-processing

When run in parallel mode the output is split into multiple family files based upon the number of parallel I/O sub-communicators that were used. Furthermore, the output in the family files is in the order that individual processors completed work. Post-processing is used to put the output into the order that a serial run would have made.

The post-processor consists of two parts, a Fortran program named Recombine Parallel Output (REPO) and a script named DOREPO. Generally the script DOREPO is run to first find the output files that are scattered across boxes and then run the REPO program to recombine the files. However, if the files are already collected the REPO program can be run alone.

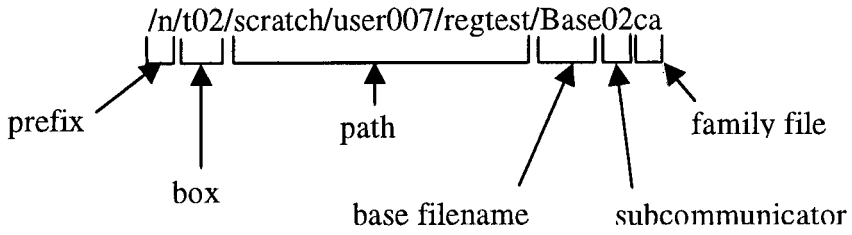

Figure 1. Path description for DOREPO.

The DOREPO script is looking for files with the filename structure shown in Figure 1. Note that the file Base01 is special. The root process of the all the processes writes it. It has more data on it than any other file, and MUST be placed first on the list of files to combine that is passed to the post-processor REPO. Output from REPO is put into a family file with "out" appended to the base filename. The DOREPO script silently deletes any old output files with the same filename to prevent REPO from appending the new output to the old output.

COMMAND LINE

The command line to the DOREPO script is

dorepo –b Base -i inrepo –d -n

where "dorepo" is the name of the script. The arguments are:

-b Base : basename of the output files to look for. The filenames have the format Base01ca where "01" is the subcommunicator number and "ca" are the family file letters. This is an optional parameter that overrides the base filename in the control file, "inrepo".

-i inrepo : the control file to determine how DOREPO runs. An optional parameter, if it is not present the default file "inrepo" is used.

-d : if –d exists, turn debugging on.

-n : if –n exists, do not run the REPO program, just gather the data files and write the REPO control input file.

DOREPO CONTROL FILE

The control file determines how the DOREPO script functions and how the post-

```
Machine     Q
Path        scratch/user007/regtest
Basename    Base
Sort        T
Outsize     75000
```

Figure 2. DOREPO control file example.

processing program is controlled. The general structure is shown in Figure 2. Note that the script is case-sensitive ONLY for the actual path and basename values. The commands are not case-sensitive, nor are the values for the machine and sort commands. Leading and trailing "/" are optional on the path value. The arguments are:

machine - can be Q, bluemountain, nirvana, or theta.

path - the path to the output files, must be the same on each box. Beginning and trailing "/" are optional.

basename - base filename.

sort - to sort or not sort the parallel cross-sections. Can be T, F, .true., .false., true, false, t, or f.

outsize – size in words of output family file members. This is equivalent to the filesize command in GIPPER.

Both sort and outsize are optional commands.

REPO CONTROL FILE

One function of the DOREPO script is to write the control file to run REPO. The control file is named "cntrl" by default. The REPO program is run by:

```
repo < cntrl
```

The REPO program expects the control file to have the commands shown in Table 2.

Table 2. Post-processor commands.

outfile bname	Base name for the output family file.
infile num	The number of input files is num, each file is specified on a separate line following this command.
sort logical	Logical to turn sorting of data on / off. Optional.
outsize	File size of the members of the output family files. Optional.
end	Control file is finished.

The commands can be in any order, except the last command has to be 'end'. Case matters, the commands must be in lower case. An example control file is shown in Figure 3. The result of REPO is a new set of output family files named bname, bnamea,

```
outfile junk
infile 2
junk01
junk02
sort .true.
outsize 250000
end
```

Figure 3. Example control file for post-processor.

bnameb, etc.

REPO will generate a single family file that contains the output in the order that a serial run would have generated. The output file will have the names "baseout, baseouta, baseoutb, ...".

---

[1] J. Abdallah Jr., R.E.H. Clark, J.M. Peek, and C.J. Fontes, **Kinetics Calculations For Near Ne-Like Ions**, J. Quant. Spectrosc. Radiat. Transfer, **51**, 1 (1994).

[2] J. Abdallah, Jr., R.E.H. Clark, D.P. Kilcrease, G. Csanak, and C.J. Fontes, **Various Applications of Atomic Physics and Kinetics Codes to Plasma Modeling**, Atomic Processes In Plasmas, Tenth Topical Conference, San Francisco, CA, January 1996, Edited by Albert L. Osterheld and William H. Goldstein, American Institute of Physics Conference Proceedings 381, Woodbury, New York, 1996.

[3] J. Abdallah Jr., R.E.H. Clark, and R.D. Cowan, **Theoretical Atomic Physics Code Development I, CATS: Cowan Atomic Structure Code**, Manual LA-11436-M, Vol. I, Los Alamos National Laboratory, December 1988.

[4] C.J. Fontes, **Development Of Relativistic Atomic Physics Codes**, Memorandum XTM: 97-90, Los Alamos National Laboratory, 1997.

[5] H.L. Zhang, C.J. Fontes, and R.E.H. Clark, **Relativistic Fine-Structure Photoionization In GIPPER**, Memorandum X-5:00-55, Los Alamos National Laboratory, April 24, 2000.

[6] J. Abdallah Jr. and R.E.H. Clark, **PARADISE Files**, Memorandum X-6: REHC-85-414, Los Alamos National Laboratory, September 22, 1986.

[7] R.E.H. Clark, J. Abdallah, Jr., and H.L. Zhang, **Interface Routines for IPCRESS Files**, LA-UR-00-3207, Los Alamos National Laboratory, 18 July 2000.